Homework 6

Christophe Hunt

March 12, 2017

Contents

1 Problem Set 1

1.1 (1) When you roll a fair die 3 times, how many possible outcomes are there?

> Sample space of a die = (1, 2, 3, 4, 5, 6)
>
> Number of outcomes = 6^3 or 216

1.2 (2) What is the probability of getting a sum total of 3 when you roll a die two times?

> Sample space of a die = (1, 2, 3, 4, 5, 6)
>
> Possible combinations equal to 3 when die_1 is rolled twice = (1,2), (2,1).
>
> There are two (2) possible combinations out of a total of 6^2 or 36 combinations.
>
> Therefore, the probability is
> $\frac{2}{36}$ = 5.56%

1.3 (3) Birthday problems

Assume a room of 25 strangers. What is the probability that two of them have the same birthday?

> We begin by calculating the probability that someone has a birthday that is not shared with another person. (e.g. the second person has a probability of 364 out of 365 of not sharing a birthday with another individual). Then we subtract that probability from one to arrive at the probability that a number of people share a birthday.

```r
prob_birth <- (0:24)
prob_birth <- 365 - prob_birth
prob_birth <- prob_birth/365
paste0("The probability that two people do not share a birthday out of ",
       length(prob_birth), " is ", percent(prod(prob_birth)))
```

```
## [1] "The probability that two people do not share a birthday out of 25 is 43.1%"
```

```r
paste0("The probability that two people do share a birthday out of ",
       length(prob_birth), " is ", percent(1 - prod(prob_birth)))
```

```
## [1] "The probability that two people do share a birthday out of 25 is 56.9%"
```

Assume that all birthdays are equally likely and equal to 1/365 each. What happens to this probability when there are 50 people in the room?

```r
prob_birth <- (0:49)
prob_birth <- 365 - prob_birth
prob_birth <- prob_birth/365
paste0("The probability that two people do not share a birthday out of ",
       length(prob_birth), " is ", percent(prod(prob_birth)))
```

```
## [1] "The probability that two people do not share a birthday out of 50 is 2.96%"
```

```r
paste0("The probability that two people do share a birthday out of ",
       length(prob_birth), " is ", percent(1 - prod(prob_birth)))
```

```
## [1] "The probability that two people do share a birthday out of 50 is 97%"
```

2 Problem Set 2

Sometimes you cannot compute the probability of an outcome by measuring the sample space and examining the symmetries of the underlying physical phenomenon, as you could do when you rolled die or picked a card from a shuffled deck. You have to estimate probabilities by other means.

For instance, when you have to compute the probability of various english words, it is not possible to do it by examination of the sample space as it is too large. You have to resort to empirical techniques to get a good enough estimate. One such approach would be to take a large corpus of documents and from those documents, count the number of occurrences of a particular character or word and then base your estimate on that.

Write a program to take a document in English and print out the estimated probabilities for each of the words that occur in that document. Your program should take in a file containing a large document and write out the probabilities of each of the words that appear in that document. Please remove all punctuation (quotes, commas, hyphens etc) and convert the words to lower case before you perform your calculations.

```
library(tm)
library(dplyr)
library(knitr)
library(RTextTools)
library(ggplot2)
library(scales)
library(DT)
library(stringr)
library(data.table)
```

```
doc <- readLines("https://raw.githubusercontent.com/ChristopheHunt/MSDA---Coursework/master/Data%20605/

doc <- Corpus(VectorSource(doc))

clean_doc <-  doc                                          %>%
              tm_map(content_transformer(tolower))         %>%
              tm_map(removeNumbers)                        %>%
              tm_map(removeWords, stopwords("english")) %>%
              tm_map(removePunctuation)                    %>%
              #tm_map(stemDocument)                          %>%
              DocumentTermMatrix()


##the entire document is too large so subsetting the matrix
freq <- as.data.frame(colSums(as.matrix(clean_doc[1:20000,])))
freq <- setDT(freq, keep.rownames = TRUE)[]
colnames(freq) <- c("word", "prob_appearance")
freq$prob_appearance <- percent(freq$prob_appearance/sum(freq$prob_appearance))
kable(head(arrange(freq,desc(prob_appearance)), n = 10))
```

word	prob_appearance
upon	0.609%
one	0.542%
new	0.507%
said	0.466%
states	0.462%
holmes	0.399%
american	0.363%

word	prob_appearance
man	0.317%
will	0.317%
government	0.305%

Extend your program to calculate the probability of two words occurring adjacent to each other. It should take in a document, and two words (say the and for) and compute the probability of each of the words occurring in the document and the joint probability of both of them occurring together. The order of the two words is not important.

```r
library(tidyr)
two_word      <-  doc                                       %>%
                  tm_map(content_transformer(tolower))      %>%
                  tm_map(removeNumbers)                     %>%
                  tm_map(removeWords, stopwords("english")) %>%
                  tm_map(removePunctuation)

new_list <- NULL
for (i in 1:length(two_word)){
  new_list[i] <- unlist(as.character(two_word[[i]][1]))
}

new_list <- paste(new_list, collapse = " ")
new_list <- gsub("\\s+"," ",new_list)
new_list <- unlist(strsplit(new_list, " "))

temp <- NULL
for (i in 1:60000) {
  temp <- append(temp, paste(new_list[i], new_list[i+1], sep = ":"))
}

new_list <- as.data.frame(new_list)

single_list <-  new_list %>%
                group_by(new_list) %>%
                top_n(n = 60000) %>%
                mutate(prob_appearance_single = n()) %>%
                mutate(prob_appearance_single = percent(prob_appearance_single/length(new_list$new_list
```

```
## Selecting by new_list
```

```r
temp     <- as.data.frame(temp)

final_list <- temp %>%
              group_by(temp) %>%
              mutate(prob_appearance_together = n()) %>%
              arrange(desc(prob_appearance_together)) %>%
              distinct(.keep_all = TRUE) %>%
              mutate(prob_appearance_together = percent(prob_appearance_together/length(temp$temp))) %>
              separate(col = "temp", into = c("single word 1", "single word 2"), ':')

final <- final_list %>%
              inner_join( single_list, by = c("single word 1" = "new_list")) %>%
              distinct(.keep_all = TRUE) %>%
              inner_join( single_list, by = c("single word 2" = "new_list")) %>%
              distinct(.keep_all = TRUE)
```

```
## Warning in inner_join_impl(x, y, by$x, by$y, suffix$x, suffix$y): joining
## character vector and factor, coercing into character vector

## Warning in inner_join_impl(x, y, by$x, by$y, suffix$x, suffix$y): joining
## character vector and factor, coercing into character vector
```

```
colnames(final) <- c("first word", "second word",
                          "prob of words together", "prob of first word",
                          "prob of second word")
kable(head(final, n = 20))
```

first word	second word	prob of words together	prob of first word	prob of second word
said	holmes	0.185%	0.646%	0.0869%
sherlock	holmes	0.167%	0.0189%	0.0869%
mr	holmes	0.118%	0.0669%	0.0869%
new	england	0.0683%	0.224%	0.0555%
st	simon	0.065%	0.0271%	0.00971%
new	york	0.065%	0.224%	0.0361%
project	gutenberg	0.05%	0.0529%	0.0211%
baker	street	0.0483%	0.00841%	0.0332%
united	states	0.0483%	0.104%	0.181%
lord	st	0.0467%	0.0235%	0.0271%
st	clair	0.0383%	0.0271%	0.0043%
let	us	0.0367%	0.0884%	0.129%
upon	table	0.0333%	0.207%	0.0532%
young	lady	0.0333%	0.116%	0.0288%
one	side	0.0283%	0.601%	0.0934%
yes	sir	0.0283%	0.127%	0.0325%
hosmer	angel	0.0283%	0.00448%	0.00672%
miss	hunter	0.0283%	0.0205%	0.0056%
mr	rucastle	0.0283%	0.0669%	0.00635%
one	two	0.0267%	0.601%	0.2%